

A VSAM/ICF Catalog White Paper

From Mainstar Software Corporation



Considerations for ICF Catalog Backup/Recovery and Day-to-Day ICF Catalog Management

If You Don't Have Mainstar's Catalog RecoveryPlus

By Ronald K. Ferguson

The two most often-heard comments about ICF Catalog management are:

"we've never had a catalog failure"

and

*"our DASD volumes never fail,
because they're all RAID
devices"*

The good news is - both of these statements can be true. The bad news is - managing an ICF catalog structure in today's complex OS/390 environment isn't that easy. Just because you don't see direct evidence of catalogs failing, and even though you have them mirrored, problems do occur, and they can have catastrophic consequences for your data center if you can't get them fixed, and fixed in a hurry! Many of today's catalog failures are subtle, and they result in logical errors across the catalog structure, or unexplained system 'hangs', or application job abends. When these types of problems occur, you might not know for weeks or months what the cause was. Worse, if catalogs are on mirrored DASD, you'll find the same problems have been carried over to the mirrored copies.

In most installations today, managers want fewer storage management staff, with less technical knowledge of operating system internals. They want their staff equipped with more automated tools, with lots of intelligence built into these tools, and a more user-friendly interface that makes the products easier to use. This is as true for the ICF catalog facility as it is for HSM, but it is often overlooked, with

people crossing their fingers and hoping that problems will never occur.

If the seriousness of this situation isn't fully appreciated, thinking that these problems never occur anymore, it could result in a serious downtime to your entire mainframe complex.

ICF catalogs are the key to accessing every single data set in your installation.

Particularly with the advent of SMS, where every data set is absolutely required to be cataloged in order to access it, if you lose a catalog you've lost production execution of the dozens (maybe hundreds) of applications that rely on that catalog. How long it takes you to restore correct access through that catalog could determine whether your business can function. **This essentially puts ICF catalog integrity, support, and management at the same level as disaster recovery planning and testing; yet many installations haven't recognized and accepted that.**

This paper deals with many of the issues surrounding backup and recovery of ICF catalogs, and identifies some reasons why every installation needs to give a high emphasis to solid, high-speed day-to-day catalog management.

Your ICF catalog environment is increasingly complex, at a rate you should be keeping your eyes on. With the huge growth in DASD storage, your installation is either constantly defining more ICF catalogs for all the new data sets on this storage, or your existing catalogs are busting at the

seams. In 1990, it would have been common that an installation had 10, 20, or 30 total catalogs; nowadays it isn't uncommon to hear of 100 or 200 catalogs. One installation that Mainstar has worked with on catalog support had 1,200 catalogs at their peak, has spent well over a year's effort in reducing this down to a current 600, and is now working to bring the number down to a more manageable 300 catalogs.

SMS is now a way of life, and it's brought wonderful new ease-of-management to data set allocations and storage management. Two factors in its fundamental design, though, have a significant impact on your ICF catalog environment, increasing its complexity and seriously increasing the recovery difficulty from any volume or catalog failure. SMS requires all data sets on SMS-managed volumes, both VSAM and non-VSAM, to be cataloged, thereby increasing the number of entries in your catalogs. SMS directs the allocation of new data sets (and data set extensions that spill over to new volumes) across any eligible volume in the data set's assigned storage group. This means more widely spread data sets that make up one application, and therefore catalogs that are 'related' to more volumes than ever before (at the same time, individual volumes are likely to be 'related' to more catalogs than ever before).

If your installation has tape libraries, there's a new catalog type, a volume catalog (or 'volcat', for short), that's going to require additional support and attention. A volcat is defined through IDCAMS exactly as an ICF user catalog is, but 'marked' as a volume catalog to the system. It contains only volume and library entries. IDCAMS provides user access to catalog entries in a volcat, for both tape libraries and tape volumes, allowing entries to be created, altered, listed, and deleted. However, IDCAMS services are intended to be used mainly as a recovery tool, to repair damaged catalogs. If IDCAMS isn't used with caution when modifying volcat entries, it can cause database (host, library manager, and tape management) conflicts. For example, IDCAMS ALTER can be used to change the use attribute of a volume in a volcat, but it does not change the library manager's inventory record or the tape

management system's database. (This warning is taken directly from the IBM manual on tape library management)

ICF catalogs, HSM CDSs, the tape management TMC, and VTOC/VTOCIXs on every DASD volume are all part of a complicated control structure that is crucial to successful access to your data. Fixing a 'break' in one but not the other, or developing a missing 'pointer' from one to another, leaves you with potential future problems in maintaining or accessing that data.

All of the technical internal details of the ICF catalog environment, such as the record layouts and field descriptions for BCS and VVDS records became OCO ('object code only') years ago with DFP 3, meaning that IBM no longer publishes any of these formats in manuals or system documentation. Nevertheless, many IDCAMS commands print BCS and VVDS records out in dump format when displaying error situations, as if this is going to somehow assist you in solving the problem. IDCAMS DIAGNOSE is famous for this, giving you a terse (and usually not very descriptive) error message, then dumping the record that is in error. This makes it extremely difficult to resolve the most serious errors between the BCS and VVDS. Unless you were lucky enough to squire away a copy of the last published IBM manual that contained these record formats before it went 'under the covers', these record dumps are of no value to you. One Catalog RecoveryPlus customer recently commented that his company decided it was more cost effective to license a catalog management tool, after he informed them it would require up to 100 hours per year of his time, allocated to de-engineering any changes that IBM made to the BCS and VVDS record formats.

Backup and Recovery

Contrary to the popular opinion that ICF catalogs never fail, let's talk about some real-life ways that ICF catalogs do break. All the old catalog management sysprogs can tell stories of the bad old days, when VSAM and ICF catalogs broke with such 'hard' failures that they couldn't even be opened, or worse, they even took MVS down with them.

Luckily, those days are in the distant past,

and nowadays our catalog failures often tend to be 'softer', causing subtle failures and errors that can go on for months without anyone knowing how serious they might be, or in mysterious system or Catalog Address Space (CAS) 'hang' or ABEND situations. They manifest in such ways as system processing through the Catalog Address Space (CAS) getting hung up, or jobs abending with IDC error messages that indicate incorrect data set access, or LISTCATs that loop or never finish. Another symptom is individual data sets that aren't accessible, even though we know they are physically present. Another is a DIAGNOSE BCS or VVDS that tells you about fatal errors on certain records, but you can't figure out how to correct them, and even if you do, you don't have commands in IDCAMS that will let you fix them. Still another is duplicate keyed records in a BCS, and since the BCS is a KSDS, duplicate records are absolutely not allowed, but due to some timing issue related to CI or CA split processing inside the BCS, a duplicate record has nevertheless occurred.

As a rule, all these types of failures are due to software errors, where possibly a recently introduced bug in catalog management causes a catalog record to be incorrectly written, or a timing glitch with cached catalog records has caused catalog management to use and then update an out-of-date record. Shared access control of a catalog that isn't correctly defined can be another frequent cause of logical catalog failures. Finally, consider just how many LPARs, across complicated sysplex structures, that you have sharing access to your ICF catalog environment, and then consider how many DASD volumes that catalog is sharing with other catalogs. When you really stop to think about that aspect of today's ICF catalogs, you begin to realize just how critical these are to your installation's successful access to your data.

How do you completely recover an ICF catalog WITHOUT forward recovery capability? One installation recently confided to us that they use nothing but EXPORT/IMPORT to backup and recover their ICF catalogs, and they've never given a thought to the out-of-date status of the catalog if they should ever have to restore the backup. They back up every catalog at midnight, once

a day, and have never performed a real-life test of how they would actually recover a catalog to current status if it were to break at, say, 4 PM the next day.

Let's see what a situation like that would require for a recovery, and then you can judge the business situation that your installation would be in should this occur. Assume your catalogs (BCSs) are all backed up at midnight. Each catalog is assigned multiple aliases that identify data set high-level qualifiers that are cataloged in it (for many installations, the number could be 50 aliases; for another, it could be 150 aliases). From midnight onward, data set allocations, deletions, and extensions to additional volumes are going on through each of the catalogs, and each results in updates to the catalog itself. In our scenario then, at 4PM, a catalog develops a problem. After careful analysis, it is determined that the only recourse is to restore the backup of the catalog that was taken at midnight. You execute an IMPORT step, which restores the catalog's records from the backup file. Obviously, though, this 'backlevels' the catalog to the previous midnight, and therefore, none of the catalog changes since midnight are now available.

To correct the backlevel situation, a series of IDCAMS DIAGNOSE commands could be set up and run (chances are, these would not already have been set up and tested ahead of time, or it would have become obvious how daunting this task is), diagnosing outwards from the restored BCS towards every VVDS that is 'related' to it ... in other words, to every volume on which there is a data set cataloged in that catalog. Then, an equal number of IDCAMS DIAGNOSE commands would also have to be run in the other direction, diagnosing from every VVDS that is related to the BCS, back towards the restored BCS. In all likelihood, this DIAGNOSE output would run to hundreds and hundreds of pages, with obtuse and terse messages for each problem found. Nevertheless, one by one, each message would have to be manually analyzed to determine the IDCAMS 'fix' that will correct the situation. How many fixes will be required depends on just how many data set defines, deletes, and extensions to new volumes have occurred in the period between backup and

restore. Complicating matters, since it's unlikely you have been running preventive maintenance DIAGNOSE commands all along, making sure that the dozens and dozens of minor errors that crop up in a catalog environment from day-to-day 'stuff' are cleaned up as you go, they will certainly now appear in the diagnose runs after the catalog recovery, cluttering up and obscuring the more important errors that need immediate attention. **Just as a shot-in-the-dark estimate, it would not be unreasonable to assume that this whole cleanup process will require a multitude of people, working flat-out for at least 8-12 hours, all the while with every application whose alias points to this catalog being halted from running. Can your installation afford that?**

But wait! There's another problem with this recovery solution that is virtually insurmountable. The recovery scenario outlined above used the DIAGNOSE command in the critical role of finding all mismatches between the DASD volumes and the catalog. DIAGNOSE is a DASD-only utility, and there is no equivalent facility to identify tape data set mismatches. In other words, updates for any tape data sets that have been cataloged and uncataloged in this BCS since midnight will not appear in the restored copy ... and there's no way to find them. Presumably, lots and lots of applications using this catalog have tape data sets of some kind that are cataloged, and this is going to be a disaster for you when you start to run the applications.

How do you completely recover an ICF catalog WITH forward recovery capability?

Catalog RecoveryPlus provides BACKUP and RECOVER commands. BACKUP operates with tremendous performance improvement compared to EXPORT, thereby enabling you to back up your catalogs more frequently if you wish to reduce your downtime in the event of a catalog failure. Some installations choose to back up every catalog once a day, others to it every four hours; or once a shift. Whatever you choose, your frequency of backup will have a significant effect on the speed and ease with which you can recover.

With EXPORT, a separate command and backup file is required for each catalog being backed up. With Catalog RecoveryPlus'

BACKUP command, you can use powerful BCS name filters that let you back up many, many catalogs in one command, and every catalog from each command is backed up and stacked to a single output file (tape or disk, your choice). This makes the commands much easier to set up, schedule, and execute, as well as easier management of the catalog backup files.

When a catalog (BCS) fails and requires a restore, you will still be restoring a backup that is 4, 8, 12, 24 hours old, and has to be 'upleveled' to the current status of the volumes and data sets that it describes. With the Catalog RecoveryPlus RECOVER command, you have an automatic forward recovery capability, using all SMF data since the backup time, to 'bring forward' the catalog. The process of 'applying' the SMF information to the catalog is performed during the restore/recovery, and at the completion of the RECOVER command, you have an up-to-date catalog. Another beauty of this is, the updating through SMF records applies to both DASD and tape data sets, so there are no 'holes' in this process. It's quick, and it's easy to run. It's also an easy process to test in advance, so when the dreaded failure does occur, you can be ready for it, with the jobs already set up and ready to run. **Odds are that a failure of this type can be recovered from, with the application back up and running, in less than one hour from the time you make the decision to restore the catalog.**

Another very good feature of a product such as Catalog RecoveryPlus is that its DIAGNOSE command is fast enough, with an incredible ease of use (compared to IDCAMS DIAGNOSE), that at the completion of the RECOVER command execution, you can easily perform a final check of the restored catalog, by running the Catalog RecoveryPlus DIAGNOSE command, with its unique ALLRELATED parameter, and every DASD data set and volume that the catalog is related to will be diagnosed. This ensures that all data sets in each (the BCS and VVDSs) correctly point to each other. If any inconsistencies are encountered, DIAGNOSE will automatically generate the necessary 'fix' commands that you can run to perform final cleanup.

Managing alias synchronization across all master catalogs in a sysplex

With the size of today's data center environments, and with the number of LPARs to process work, the overall number of master catalogs is much greater than ever before. Alias pointers to all connected user catalogs are physically contained in each of the master catalogs. In many installations, the master catalogs that share DASD across a complex usually have the same set of user catalogs connected to them, with a totally matching, synchronized, set of aliases for all the user catalogs. For correct day-to-day operation of these OS/390 systems, it's imperative that the aliases in all master catalogs remain synchronized.

With standard IDCAMS support, there is no easy way to maintain this synchronization, other than to print out the alias records from each master catalog, and then through meticulous listing analysis, identify missing or superfluous alias entries in each master and manually create the DEFINE/DELETE ALIAS commands to correct all out-of-synch conditions. This would have to be an ongoing, routine process, and one that could cost a lot of time and resource if it is done, and something that can cause a tremendous amount of job failure grief if it isn't done.

An easier way to manage this situation is with Catalog RecoveryPlus, where its DIAGNOSE ALIAS command can analyze the aliases across master catalogs and update them to be in synch. Because it's important to know which master catalog is the 'control catalog' for this, the best procedure is to purposely identify one master catalog to which you will perform all routine alias maintenance, and then point the Catalog RecoveryPlus DIAGNOSE ALIAS command to that catalog for synchronization to all other master catalogs in the complex.

Merging and splitting catalogs within short time windows

The term 'merging' and 'splitting' catalogs really refers to one process — moving catalog entries from one catalog and putting them in another. This is not a copy process, but rather

a move process, as the catalog records are removed from the source catalog as they are written to the target catalog. If an entire alias group of data set entries is moved, it's important for the related alias to now 'point' to the new catalog.

IDCAMS has a MERGECAT keyword on the REPRO command, where catalog entries are moved (not copied), one at a time, and the individual VVDS updates take place at the time the BCS entry is moved. This is an incredibly slow command to execute, and suffers from inherent operational safety issues with its processing method. By updating the VVDS back-pointers in series with the catalog entry moves, a VVDS somewhere on the system has to be opened up, read from, then written to, and then closed, for each BCS entry that is moved. With this simplistic processing logic, if any interrupt in processing occurs during the MERGECAT operation, you either have to manually set up the command to run backwards, or you have to hope the command can be restarted and then successfully completed. There is no facility to explicitly state that you want to 'undo' the merge operation up to the point of failure, and there is no logic or keyword specification that tells the processor you are attempting to restart an interrupted MERGECAT. You simply have to hope that the command is smart enough to figure out that a backward merge is being performed, or that a restart is being attempted, and that it will somehow know what to do, and then be able to do it.

Additionally, the IDCAMS MERGECAT facility command (and other third-party vendor products on the market) do not update aliases in the master catalog(s) for any complete 'levels' of data set entries moved from the source to target catalog. Remember the installation mentioned above — the one that had 1,200 catalogs at their peak, has spent well over a year's effort in reducing this number down to a current 600, and is now working to bring this down to 300? Well, consider how many data set aliases were in master catalogs connected to those user catalogs and you get an idea of how much more manual work you have in this process. It isn't at all unusual for user catalogs in today's environment to have 50, 100, or many more

aliases assigned to them, and if your reason for running MERGECAT is to reduce the total number of catalogs, you'll find yourself manually creating and running a DEFINE ALIAS command for each and every one of those aliases ... and then running these commands against every master catalog in your complex.

To illustrate the problem of a slow-running IDCAMS MERGECAT, let's consider again the installation that was reducing 1,200 catalogs to 600. These were all production catalogs, and may have had aliases assigned to them that are scattered all over the spectrum of production applications. Most installations would have serious reservations about running a MERGECAT on a production catalog, while that catalog is in production use with some application job streams, but with some applications quiesced for which a LEVEL specification MERGECAT is to be run. Therefore, they would most likely want to lock down the entire catalog while the MERGECAT is being performed. So, in order to run the necessary MERGECAT commands, standalone time has to be carefully scheduled, two hours here, two hours there, across a wide period of time. Now the problem for the person doing this is, his block of time was very specific, and production *had* to resume right on schedule at the end of his catalog merging time, with no margin for error. Using IDCAMS MERGECAT, therefore, he had no choice but to plan his MERGECATs in such a way that he always had sufficient time remaining at the end of his standalone test block, so that he could completely 'undo' a partially complete MERGECAT that experienced problems. Remember, though, that MERGECAT updates VVDSs as it moves source-to-target BCS entries, so if there was an interruption, his only recourse was if he could either run the MERGECAT backwards or get it to restart and continue to end of job. Here's the hitch: just taking a backup of the catalog before you start the MERGECAT isn't going to be good enough, since if the MERGECAT fails part way through it, restoring the catalog backup is still going to leave you with all those updated VVDSs scattered out across an untold number of volumes. Backing up and then restoring those volumes just isn't an option, considering how many volumes we're talking about here.

Keeping the ICF Catalog environment 'clean' through a frequent DIAGNOSE plan

As mentioned above in the BACKUP and RECOVER discussion, finding out that a catalog has dozens, possibly hundreds, of relatively minor errors, at the very time you're trying to recover from a major error in that catalog just isn't a good idea. It will seriously lengthen the time necessary to get back up and running, and throughout that time, every application (and your online systems) that needs to use that catalog to access data sets is waiting to begin execution.

IDCAMS DIAGNOSE is the command provided by ICF catalog support for identifying errors in your catalog structure, enabling you to create the necessary fixes that will clean up any problems. The trouble with IDCAMS DIAGNOSE, though, is that it's cumbersome and kludgy to use, slow in execution, and absolutely produces tons of output reports that you need to plow through. While this is OK for a one-off situation, it isn't feasible for preventive maintenance of your overall catalog structure. Sure, the IDCAMS DIAGNOSE command will perform a diagnostic function from a BCS outwards to many VVDSs, and also diagnose from a VVDS outwards to many BCSs ... the problem is, you have to specifically state the 'to' VVDSs and BCSs in each command, and frankly, in an SMS environment (where allocations can occur across an entire storage group), it just isn't feasible to keep up with this.

Take, for example, a VSAM cluster for a KSDS file that has somehow become uncataloged ... in other words, the index and data components are physically present on their volume, but the cluster record is missing from the appropriate BCS catalog. You run a DIAGNOSE VVDS, comparing it to the appropriate BCS. In the output report from this, you receive separate error messages that the cluster record is missing for both the index and data component. The trouble is, DIAGNOSE does not correlate that the two data set components being reported on are from the same cluster, and the output for this can be scattered a dozen (or a hundred) pages apart, depending on how many other data set

problems are being reported on. It's your responsibility to manually (most likely, laboriously, building a hand-written correlation table) interpret all of these errors, and then relate them together. When that's all finished, you then have to manually build the DEFINE CLUSTER RECATALOG commands. If the data set is multi-volume, you also have to identify all of the volumes for the data set, by matching up component error messages across DIAGNOSE reports for the other volumes. If the cluster has alternate indexes, you again have to manually correlate all of this together, building up the DEFINE ... RECATALOG commands for all AIXs, as well as PATHS. This is a *very* time-consuming and laborious effort, and that's very likely to be the most common reason most installations just never get around to doing this on a regular basis.

With Catalog RecoveryPlus, you have a DIAGNOSE capability in *all* directions (BCS-VVDS-VTOC), with the ability to specify large numbers of BCSs and VVDSs through pattern mask filters, plus the ability to specify the ALLRELATED parameter, so that every included BCS and VVDS is diagnosed outward to every related VVDS and BCS. Most importantly, fix commands are automatically generated and stored in a 'fix' file that you can then review and edit before submitting to correct all the problems. The speed of execution on this is such that it can be run on a regular basis, maybe once a week, against every BCS and VVDS in the shop, enabling you to stay on top of errors that are creeping into the ICF catalog environment. That way, you can also begin to get a handle on what's causing these errors to occur, and make some attempts to eliminate the root problem.

Merging workloads, sysplexes, and data centers - how does this affect volumes and ICF catalogs?

Why do I need a BCS Rename facility?
The business world is full of corporate mergers, spin-offs, and outsourcing contracts that are starting or ending. These instances certainly must be keeping the world of IT really jumping. With the hundreds of thousands, maybe millions, of data sets under

management, it's no wonder that we have thousands of volumes, and hundreds of ICF catalogs in many installations.

If any one of the above-mentioned corporate structure changes, or data center realignments occur, all of the best data set naming standards, volume name standards, and ICF catalog naming standards go out the window. How do you make these changes when all you have is a utility product (IDCAMS) that was created for by-gone days? Something as simple as changing the name of a catalog is essentially impossible with IDCAMS and OS/390 utilities, as it would involve the unloading of every data set cataloged in the catalog, then deleting the catalog and redefining it with a new name, and finally, defining and restoring all of the data sets. This is a job of such magnitude that it isn't even considered something to think about. For that reason, installations have an absolute hodge-podge of catalog naming conventions.

Catalog RecoveryPlus has a BCS NEWNAME facility in its RECOVER command, enabling you to effortlessly rename a catalog as part of a restore/recovery operation. In other words, this could simply be done at the time you reorganize a catalog. It's quick, easy, and can be run at will.

Why do I need the capability to quickly and effortlessly 'clip' volsers?

For all the reasons mentioned above for BCS NEWNAME, many installations also need the ability to rename DASD volsers in a quick, easy manner. There is no easy, fast facility in OS/390 to change a volser, and as a result, it isn't done very often for volumes that already contain data. It would be very helpful, though, in maintaining volume serial naming standards, if there was a way to do it effortlessly and quickly. With standard OS/390 facilities, the only way you can change a volser is to manually unload every data set off the volume (i.e., copy to tape, then delete the disk copy), then clip the volume by running ICKDSF, then define and reload each of the data sets. If there happens to be a catalog on the volume to be clip'd, you also need to delete the catalog with the RECOVERY operand, so that none of its data sets are also deleted, then unload all

data sets from that catalog that are physically present on the volume ... as above.

Catalog RecoveryPlus has a SUPERCLIP command that allows you to clip a DASD volume without having to copy the data from the volume. Without all the hassle described above, you only need to quiesce the volume for this to work. SUPERCLIP internally clips the volser for you (by executing ICKDSF under the covers), then automatically updates all VVDS/VTOC/VTOCIX and catalog information with the new volume serial number. The DASD volume can also have a catalog physically resident on it, and all data set information in that catalog and system will be automatically updated.

You can even clip a sysres volume, with the restriction that you must execute SUPERCLIP from another system whose sysres volume(s) are not the ones being changed. Volumes that cannot be SUPERCLIPed include HSM ML1 volumes, and any volume where information about the volser is contained in a table or 'catalog' that is not part of the ICF or VTOC environment.

**Ronald K. Ferguson - Founder,
President & CEO of Mainstar Software
Corporation**

Ron Ferguson has a technical background in large-scale OS/390 systems. As a software instructor for 15 years, he has presented over 600 courses on VSAM, and is recognized worldwide as a data storage and recovery expert. Ferguson travels widely, meeting with customers and presenting at national and international conferences.